



**You have downloaded a document from
RE-BUŚ
repository of the University of Silesia in Katowice**

Title: The program-substitution in algorithmic logic and algorithmic logic with non-deterministic programs

Author: Andrzej Biela

Citation style: Biela Andrzej. (1984). The program-substitution in algorithmic logic and algorithmic logic with non-deterministic programs. "Bulletin of the Section of Logic" (Vol. 13, no. 2 (1984), s. 69-72).



Uznanie autorstwa - Użycie niekomercyjne - Bez utworów zależnych Polska - Licencja ta zezwala na rozpowszechnianie, przedstawianie i wykonywanie utworu jedynie w celach niekomercyjnych oraz pod warunkiem zachowania go w oryginalnej postaci (nie tworzenia utworów zależnych).



UNIwersYTET ŚLĄSKI
W KATOWICACH



Biblioteka
Uniwersytetu Śląskiego



Ministerstwo Nauki
i Szkolnictwa Wyższego

Andrzej Biela

THE PROGRAM-SUBSTITUTION IN ALGORITHMIC LOGIC AND ALGORITHMIC LOGIC WITH NON-DETERMINISTIC PROGRAMS

This note presents a point of view upon the notions of program-substitution which are the tools for proving properties of programs of algorithmic logics [5], [3] being sufficiently strong and universal to comprise almost all previously introduced theories of programming, and the so-called extended algorithmic logic [1], [2] and algorithmic logic with non-deterministic programs [4].

It appears that the mentioned substitution rule allows us to examine more deeply algorithmic properties of terms, formulas and programs. Besides the problem of Post-completeness and structural completeness of algorithmic logics strengthened additionally by the rule of substitution is raised.

For $i \in \{1, 2, 3\}$, L_i denote the language defined in [3], [1] and [4], respectively. In turn, $T, F, S, FS_i, FST_i, FSF_i$ are sets of classical terms, open' formulas, substitutions, programs, terms and program-formulas respectively.

By E_i we denote the set of all elementary formulas. We put $At_i = V_0 \cup \{1, 0\} \cup (E_i \cap F)$. By $C_{A_i R_i}$ we denote the consequence operations of the algorithmic logics defined in [3], [1], [4] respectively. We shall write $X \vdash_i$ instead of $\alpha \in C_{A_i R_i}(X)$ and $X = \emptyset$ will be omitted.

Let g be any one-one mapping of the set $V \cup V_0$ into $V \cup V_0$ such that $g(V) \subseteq V$ and $g(V_0) \subseteq V_0$. It is clear that any such mapping can be extended to an endomorphism g' defined on $T \cup F$. If s is a substitution and f is a mapping from T into T and from F into F , then by $f(s)$ we denote the substitution obtained from s by exchanging all expressions of the form $x_k, \tau_k, a_j, \alpha_j$ by $f(x_k), f(\tau_k), f(a_j), f(\alpha_j)$, respectively.

ε_g^i is the set of all endomorphism on F such that $e(1) = 1, e(0) = 0$ and $e(s\rho(\tau_1, \dots, \tau_n)) = g'(s)e(\rho(\tau_1, \dots, \tau_n))$ for every $\rho(\tau_1, \dots, \tau_n) \in E_i \cap F$ and $s \in S$.

For any $e \in \varepsilon_g^i$, let e_g be an endomorphism on F such that $e_g(\alpha) = g(\alpha)$ for every $\alpha \in V_0$, and $e_g(\alpha) = e(\alpha)$ for every $\alpha \in At_i - V_0$.

For any program K and any function $e \in \varepsilon_g^i$ we define K_g^e as follows: if $K = [x/\tau_1, \dots, x_n/\tau_n, a_1/\alpha_1, \dots, a_m/\alpha_m]$ then $K_g^e = [g(x_1)/g'(\tau_1), \dots, g(x_n)/g'(\tau_n), g(a_1)/e_g(\alpha_1), \dots, g(a_m)/e_g(\alpha_m)]$; if K is one of the form $\circ[MN], \vee[\delta MN], *[\delta M], [M \cup N]$, then K_g^e is of the form $\circ[M_g^e N_g^e], *[e_g(\delta)M_g^e N_g^e], *[e_g(\delta)M_g^e], [M_g^e \cup N_g^e]$ respectively to the language L_i .

Let \bar{e}_g be an endomorphism on FSF_i satisfying the following conditions: $\bar{e}(\alpha) = e_g(\alpha)$ for every $\alpha \in F$, $\bar{e}_g(K\alpha) = K_g^e \bar{e}_g(\alpha)$, $\bar{e}_g(\bigcup K\alpha) = \bigcup K_g^e \bar{e}_g(\alpha)$, $\bar{e}_g(\rho(\tau_1, \dots, \tau_n)) = \bar{e}_g(\chi(\rho(\tau_1, \dots, \tau_n)))$ for $i = 1$, $\bar{e}_g(\exists_x \alpha) = \exists_{g(x)} \bar{e}_g(\alpha)$ for $i \in \{2, 3\}$. Then for $i = 3$, we put $\bar{e}_g(\bigcap K\alpha) = \bigcap K_g^e \bar{e}_g(\alpha)$, $\bar{e}_g(\forall_x \alpha) = \forall_{g(x)} \bar{e}_g(\alpha)$ and $\bar{e}_g(DK\alpha) = DK_g^e \bar{e}_g(\alpha)$ for any $D \in \{\nabla, \nabla \bigcup, \nabla \bigcap, \nabla, \Delta \bigcup, \Delta \bigcap\}$.

For any expression w , $\mathcal{V}(w)$ denotes the set of all variables of w . For a couple of functions $\langle f, f' \rangle$ such that $f : T \cup F \rightarrow T \cup F$, f restricted to V_0 is one-one mapping from V_0 into V_0 , $f' : F \rightarrow F$ and for every $\alpha \in FSP_i$ such that $\mathcal{V}(\alpha) \cap V_0 = \{a_1, \dots, a_m\}$ we put $s^\alpha = [f(a_1)/f'(a_1), \dots, f(a_m)/f'(a_m)]$. If $\mathcal{V}(\alpha) \cap V_0 = \emptyset$, then we put $s^\alpha = []$. Further we shall say that s^β is designated by $\langle f, f' \rangle$.

For any $e \in \varepsilon_g^i$ we define e^g as follows:

$$e^g(\alpha) = e(\alpha) \text{ for } \alpha \in F \text{ and } e^g(\alpha) = \begin{cases} s^{\chi(\alpha)} \bar{e}_g(\alpha) & \text{for } i = 1 \\ s^\alpha \bar{e}_g(\alpha) & \text{for } i \in \{2, 3\} \end{cases} \text{ for } \alpha \in FSF_i - F$$

A function \bar{e} defined on FSF_i is called a program-substitution ($\bar{e} \in \varepsilon^i$) if $\bar{e} = e^g$ for some g and $e \in \varepsilon_g^i$.

LEMMA 1. For every open formula α and program-formula β and for every $e \in \varepsilon_g^i$, $s \in S$ the following properties hold:

- a. $g(V_0) \cap \mathcal{V}(e(E_i \cap F)) = \emptyset$,
- b. $\overline{s_g^e e_g(\alpha)} = e_g(\overline{s\alpha})$,
- c. If $V_0 \cap \mathcal{V}(\alpha) \subset \mathcal{V}(\beta)$, then $\overline{s^\beta e_g(\alpha)} = e(\alpha)$ where s^β is designated by $\langle g, e \rangle$,
- d. For every $\gamma \in FSF_i$ and for every $y \in V$, if $y \notin \mathcal{V}(\gamma)$, then $g(y) \notin \mathcal{V}(\bar{e}_g(\gamma))$.

THEOREM 1. *Algorithmic logic is closed under program-substitution, i.e. $\bar{e}(C_{A_i R_i}(\emptyset)) \subseteq C_{A_i R_i}(\emptyset)$ for every $\bar{e} \in \varepsilon^i$.*

By r_* we denote the substitution rule, that is, $\langle \{\alpha\}, \beta \rangle \in r_*$ iff $\beta = \bar{e}(\alpha)$ for some $\bar{e} \in \varepsilon^i$. Let $R_i^* = R_i \cup \{r_*\}$. Obviously, $C_{A_i R_i^*}(\emptyset) = C_{A_i R_i}(\emptyset)$.

LEMMA 2. *For every $\alpha, \beta \in FSF_i$ and $e \in \varepsilon_g^i$: if $\mathcal{V}(\alpha) \cap V_0 \subseteq \mathcal{V}(\beta)$, then $\vdash_i s^\beta \bar{e}_g(\alpha) \leftrightarrow s^\alpha \bar{e}_g(\alpha)$ for $i \in \{2, 3\}$ and for $i = 1$ instead $\mathcal{V}(\alpha)$, s^α we must write $\mathcal{V}(\chi(\alpha))$, $s^{\chi(\alpha)}$, where $s^\beta, s^\alpha, s^{\chi(\alpha)}$ are designated by $\langle g, e \rangle$.*

THEOREM 2. *For every $\bar{e} \in \varepsilon^i$ and $\alpha, \beta \in FSF_i$: $\vdash_i \bar{e}(\alpha \cdot \beta) \leftrightarrow (\bar{e}(\alpha) \cdot \bar{e}(\beta))$ for $\cdot \in \{\rightarrow, \cdot, +\}$ and $\vdash_i \bar{e}(\sim \alpha) \leftrightarrow \sim \bar{e}(\alpha)$.*

THEOREM 3. *The consequence $C_{A_i R_i^*}$ is Post-incomplete.*

A rule r is called structural if $\langle \bar{e}(X), \bar{e}(\alpha) \rangle \in r$ for every sequent $\langle X, \alpha \rangle \in r$ and $\bar{e} \in \varepsilon^i$.

For $i = 1$ we introduce the notion of algorithmic structural completeness which slightly differs from the known examination concerning the property of structural completeness [6]. If $X \subset FSF_i$ and $K \in FS_i$, then by KX we shall denote the set of all formulae of the form $K\alpha$ for any $\alpha \in X$.

For any $D \subset FS_1$ we shall say that the rule r is D -admissible in a consequence C if for every $\langle X, \alpha \rangle \in r$ and $K \in D$, $KX \subset C(\emptyset)$ implies $K\alpha \in C(\emptyset)$.

If $D = S$, then instead of saying that the rule r is S -admissible we shall say that the rule r is program-admissible.

Now we define the set $J \subset FSF_1$ as follows: $\alpha \in J$ iff there exists an open formula β such that $\vdash_1 \alpha \leftrightarrow \beta$.

A rule r is finitary if for every $\langle X, \alpha \rangle \in r$ the set X is finite and $X \cup \{\alpha\} \subset J$. We shall say that the consequence C in L_1 is algorithmically structurally complete if every structural, finitary and program-admissible rule r of C is derivable in it.

THEOREM 4. *The consequence $C_{A_1 R_1^*}$ of algorithmic logic is algorithmically structurally complete.*

For $i \in \{2, 3\}$, the problem of algorithmic structural completeness is open.

References

- [1] L. Banachowski, *Investigations of properties of programs by means of the extended algorithmic logic I*, **Fundamenta Informaticae**, Vol. I, No. 1 (1977), pp. 93–119.
- [2] A. Kreczmar, *Effectivity problems of algorithmic logic*, *ibid.* Vol. I, No. 1 (1977), pp. 19–32.
- [3] G. Mirkowska, *Algorithmic logic and its applications in the theory of programs I, II*, *ibid.*, Vol. 1, No. 1 (1977), pp. 1–17 and No. 2 (1977), pp. 147–165.
- [4] G. Mirkowska, *Model existence theorem in algorithmic logic with nondeterministic programs*, *ibid.*, Vol. III, No. 2 (1980), pp. 157–170.
- [5] A. Salwicki, *Formalized algorithmic language*, **Bulletin de l'Académie Polonaise des Sciences. Série des Sciences Mathématiques, Astronomiques et Physiques** 18 (1970), pp. 227–232.
- [6] W. A. Pogorzelski, *Structural completeness of the propositional calculus*, *ibid.*, Vol. 19 (1971), pp. 349–351.

*Department of Mathematics
Silesian University
Katowice, Poland*